

Constraints (Primary Key / Foreign Key)

Constraints are used to:

1. Define the primary key for the table.
 - Implemented via the definition of the primary key constraint.
2. Define relationships between tables.
 - Implemented via the definition of the foreign key constraint.

Constraints can be defined:

1. When the table is initially created, as part of the Create Table statement or,
2. For an existing table, via the Alter Table statement.

We use the Constraint keyword to define a constraint. A single column can have multiple constraints, each with its own constraint keyword definition. Each constraint must have a unique name. The constraint name is used in SQL messages, so use descriptive constraint names. We will use a prefix, as part of the constraint name, to identify the type of constraint:

Prefix	Constraint Type
--------	-----------------

PK	Primary Key
FK	Foreign Key
CK	Check
DF	Default
UQ	Unique

A constraint can be defined as a column level constraint (applies to one column only) or possibly as a table level constraint (applies to many columns in the table).

Note: constraint names in this document are shortened for readability and should not be considered appropriate names.

Primary Key Constraint

When using a normalized database design, all tables must have a primary key constraint. Any column that acts as a primary key must be

defined as a **not null** column. The DBMS will ensure that all rows in the table have a unique value for primary key columns.

When a primary key constraint is defined, PostgreSQL automatically creates a unique index for the column(s) acting as the primary key.

Syntax

Column constraint syntax:

[Constraint ***constraint_name***] Primary Key

Table constraint syntax:

[Constraint ***constraint_name***]
Primary Key (***col_name*** [, ***col_name2*** [, . . . ***col_name32***]])

Example:

To define the StudentID column as a primary key in the Student table:

```
Create Table Student
(
    StudentID  char (9)      not null
                                Constraint PK_Student Primary Key,
    LastName   varchar (20)   not null,
    FirstName  varchar (15)   not null
);
```

This example demonstrates a column level primary key constraint definition.

A table level primary key constraint is used when the table has a composite key (multiple columns acting as the primary key).

Example:

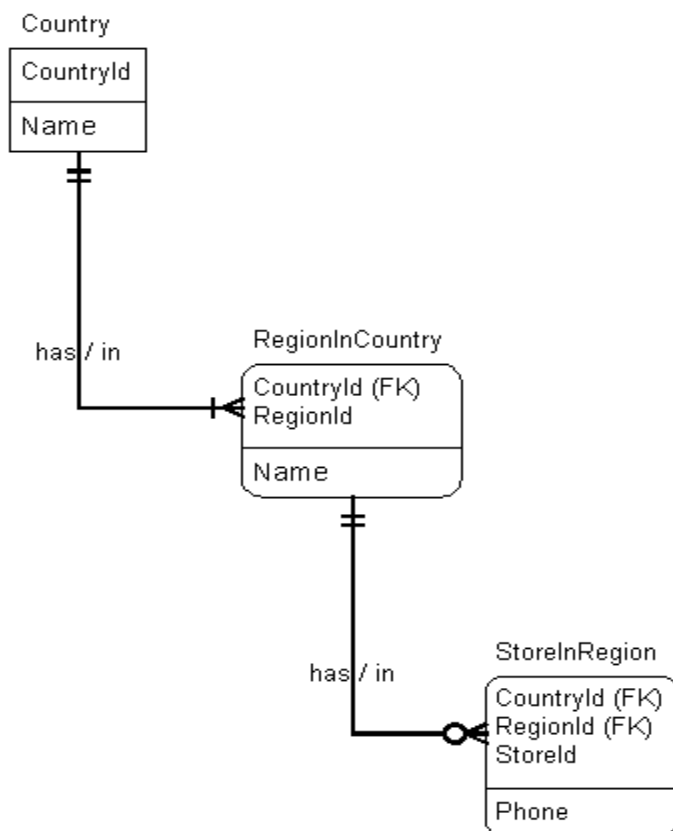
To define the StudentID and CourseID columns as the primary key for the Marks table:

```
Create Table Marks
(
    StudentID  char (9)      not null,
    CourseID   char (6)      not null,
    Mark       integer       null,
    Constraint PK_Marks Primary Key
    (StudentID, CourseID)
);
```

);

Foreign Key Constraint

The foreign key constraint defines a relationship between rows (usually in different tables). In addition, the foreign key constraint defines a parent-child relationship between the tables involved. The table with the foreign key acts as the child table in the relationship. The table with the associated primary key acts as the parent table in the relationship. Consider the following tables for a chain of stores that operates internationally.



CountryID and StoreID are unique. RegionID is unique within Country (i.e. the first region in any country is region 1).

In the RegionInCountry table (child) the foreign key, CountryID, relates rows to the parent table, Country. A column level foreign key constraint, for the CountryID column in the RegionInCountry table, implements this relationship.

In the StoreInRegion table (child) the foreign key, CountryID and RegionID, relates rows to the parent table, RegionInCountry. A table level foreign key constraint, CountryID and RegionID columns, in the StoreInRegion table, implements this relationship.

The **foreign key constraint** defines referential integrity between two tables. Operations that affect the tables and/or the data stored within the tables must comply with referential integrity. This affects:

1. Dropping tables.
2. Creating tables.
3. Inserting/Updating/Deleting rows in tables.

The bottom line for referential integrity is that a child row/table cannot exist without its associated parent row/table. Therefore:

1. You must Drop a child table BEFORE you Drop the associated parent table.
2. You must Create a parent table BEFORE you Create the associated child table.
3. The value of a column acting as a foreign key must be:
 - A value that exists as a primary key in the associated parent table, or
 - NULL

If we were to create a script file to drop the existing definition of the Store tables and recreate them, we would sequence the commands in the following order:

```
Drop Table StoreInRegion;  
Drop Table RegionInCountry;  
Drop Table Country;;  
  
Create Table Country (. . .);  
Create Table RegionInCountry (. . .);  
Create Table StoreInRegion (. . .);
```

A column acting as a foreign key can be defined as a NULL or as a NOT NULL column and must have the same datatype as its associated

primary key. The DBMS will NOT automatically create an index for a foreign key.

Syntax:

Column constraint syntax:

[Constraint ***constraint_name***
References ***ref_table*** (***ref_column***)

Table constraint syntax:

[Constraint ***constraint_name***
Foreign Key (***column_name*** [, . . . ***column_name32***])
References ***ref_table*** (***ref_column*** [, . . . ***ref_column32***])

- ***column_name*** identifies the column(s) acting as the foreign key
- ***ref_table*** identifies the parent table
- ***ref_column*** identifies the primary key in the associated parent table

Example:

To define the foreign key constraint for the RegionInCountry table:

Create Table RegionInCountry

```
(
    CountryID integer not null
                Constraint FK_RegionInCountry_Country
                References Country (CountryID),
    RegionID integer not null,
    Name varchar (100) not null,
    Constraint PK_RegionInCountry_CountryID_RegionID
                Primary Key (CountryID, RegionID)
);
```

This demonstrates the use of a column level foreign key constraint. Note that a single column (e.g. CountryID) can act as both a foreign key and part of a primary key.

Example:

To define the foreign key constraint for the StoreInRegion table:

Create Table StoreInRegion

```
(
    CountryID integer not null,
    RegionID integer not null,
    StoreID integer not null,
    Phone varchar (100) not null,

    Constraint PK_CountryId_RegionId_StoreID
        Primary key (CountryID, RegionID, StoreID),

    Constraint FK_StoreInRegion_RegionInCountry
    Foreign Key (CountryID, RegionID)
    References RegionInCountry (CountryID,
    RegionID)
);
```

This demonstrates the use of a table level foreign key constraint. The foreign key constraint applies to two columns, hence the use of the table level constraint. It is technically valid to define a single-column foreign key as a table constraint, but it is considered better practice to define it as a column-level constraint.